Revista Eletrônica de Sistemas de Informação
ISSN 1677-3071

v. 13, n. 2

May-Ago 2014 - special issue on FOSS

doi:10.5329/RESI.2014.1302

Editor's space

Editorial
Carlos Denner dos Santos Jr.

Essays

RIO IN THE GLOBAL WORLD OF SOFTWARE
Yuri Takhteyev

Free software

THE ALGORITHMIC AUTOREGULATION SOFTWARE DEVELOPMENT METHODOLOGY
Renato Fabbri, Ricardo Fabbri, Vilson Vieira, Daniel Penalva, Danilo Shiga, Marcos Mendonça, Alexandre Negrão, Lucas Zambianchi, Gabriela Salvador Thumé

FRENCH SOFTWARE POLITICS: ACTIVISM AND THE DYNAMICS OF GLOBALIZATION FROM BELOW
Pierre-Amiel Giraud, Sara Schoonmaker

O USO DO SOFTWARE LIVRE NA NA APRENDIZAGEM COLABORATIVA: LIMITES E POSSIBILIDADES DO PROGRAMA "UM COMPUTADOR POR ALUNO"
Rafaela da Silva Melo, Ana Beatriz Gomes Pimenta de Carvalho

HISTORICAL ANALYSIS OF MESSAGE CONTENTS TO RECOMMEND ISSUES TO OPEN SOURCE SOFTWARE CONTRIBUTORS
Igor Fabio Steinmacher, Igor S Wiese, Andre Luis Schwerz, Rafael Liberato Roberto, João Eduardo Ferreira, Marco Aurélio Gerosa

SENTIMENT ANALYSIS OF FREE/OPEN SOURCE DEVELOPERS: PRELIMINARY FINDINGS FROM A CASE STUDY
Athanasios-Ilias Rousinopoulos, Gregorio Robles, Jesús M. González-Barahona

ESTUDIO PRELIMINAR DE LAS EMPRESAS QUE MÁS CONTRIBUYEN AL DESARROLLO DE ANDROID
Sergio Raul Montes León, José David Ramos Benalcazar, Hernán Montes León

SENTIMENT ANALYSIS OF FREE/OPEN SOURCE DEVELOPERS: PRELIMINARY FINDINGS FROM A CASE STUDY

ANÁLISE DE SENTIMENTOS DE DESENVOLVEDORES DE SOFTWARE LIVRE: ACHADOS PRELIMINARES DE UM ESTUDO DE CASO

(artigo submetido em agosto de 2013)

Athanasios-Ilias Rousinopoulos
GSyC/LibreSoft – Universidad Rey Juan Carlos (URJC)
athanrous@gmail.com

Gregorio Robles
GSyC/LibreSoft – Universidad Rey Juan Carlos (URJC)
grex@gsyc.urjc.es

Jesús M. González-Barahona
GSyC/LibreSoft – Universidad Rey Juan Carlos (URJC)
jgb@gsyc.es

ABSTRACT

Software development is a human intensive activity. And as such, how developers face their tasks is of major importance. In an environment such as the one that is common in FOSS (free/open source software) projects where professionals (i.e., paid developers) share the development effort with volunteers, the morale of the development and user community is of major importance. In this paper, we present a preliminary analysis using sentiment analysis techniques to a FOSS project. We therefore mine the mailing list of a project and apply these techniques to the most relevant participants. Although the application is at this time limited, we hope that this experience can be of benefit in the future to determine situations that may affect the developers or the project, such as low productivity, developer abandonment, project forking, etc.

Key-words: openSUSE; FLOSS; sentiment analysis; software development; software repository mining; mailing lists; developer productivity; natural language processing.

RESUMO

O desenvolvimento de software é uma atividade intensive em esforço humano. Assim, a forma como os desenvolvedores encaram suas tarefas é de suam importância. Em um ambiente como o usual em projetos de FOSS (*free/open source software*) em que profissionais (desenvolvedores pagos) compartilham os esforços de desenvolvimento com voluntários, a moral da comunidade de desenvolvedores e usuários é fundamental. Neste artigo, apresentamos uma análise preliminary utilizando técnicas de análise de sentimentos realizada em um projeto de FOSS. Para isso, executamos a mineração da lista de endereços eletrônicos de um projeto e aplicamos as técnicas propostas aos participantes mais relevantes. Embora a aplicação seja limitada, no momento atual, experamos que essa experiência possa ser benéfica no future para determiner situações que possam afetar os desenvolvedores ou o projeto, tais como baixa produtividade, abandono do projeto ou bifurcação do projeto, entre outras.

Palavras-chave: openSUSE; FLOSS; análise de sentimentos; desenvolvimento de software; mineração de repositórios de software; listas de endereços eletrônicos; produtividade do desenvolvedor; processamento de linguagem natural.

1 INTRODUCTION

The motivation of why FOSS (free/open source software) developers contribute to software projects has been a matter of study since the early 2000s (GHOSH *et al.*, 2002; HERTEL, 2003). However, to the knowledge of the authors there are no techniques to ascertain how developers (and users) feel about the software they are developing (using) and how these feelings might affect the project. It is clear, that bad feelings may be the first step to low productivity, to abandon the project, to create conflicting situations, and even to fork the project. This methodology is known in the academic world as "text sentiment analysis" and can be defined as *a text mining technique to analyze the sentiment of the writer or to the topic written about*. Furthermore sentiment analysis may use machine learning techniques.

Hence, in this work we have as main goal to analyze the evolution of the sentiment of developers of the openSUSE Factory, a FOSS project, during a time span of 27 months where three releases had been released, the latest one with a three-month delay. We will therefore use a general methodology that can be applied to any FOSS project, which is based in the analysis of the e-mails sent to the mailing list of the project.

The rest of the paper is structured as follows: in the next section, related work is presented. Then we introduce our methodology and devote a section to present the case study that has been selected. Section 4 provides the results of applying the methodology to the case study. Finally, conclusions are drawn and ideas for further research are discussed.

2 RELATED WORK

In the last years, researchers have been working with sentiment analysis in many aspects. There are many cases where scientists do study the multilingual sentiment analysis in more details (BADER *et al.*, 2011, BAL *et al.*, 2011, GÎNSCA *et al.*, 2011, BALAHUR AND TURCHI, 2012, BOYD-GRABER AND RESNIK, 2010). Lately, a majority of the research has been focused on the sentiment analysis on the web. In terms of sentiment analysis on the web, scientists focus more on social media sentiment analysis (PALTOGLOU AND THELWALL, 2012), e.g. Twitter (LEE AND ANAM-DONG, 2012; SAIF, HE AND ALANI, 2012, WANG *et al.*, 2012) than on more traditional platforms such as forums and mailing lists. Despite that fact, researchers from Stanford University (California, USA) performed a study in the field of email sentiment analysis (HANGAL AND LAM, 2011). Although the study shows the impact of the real sentiment which is expressed in personal communication, the study domain is more generic and not focused in any FOSS development case, specifically.

For finding these techniques in mailing lists, we have to go to a study hosted by the U.S. National Center for Biotechnology Information (BEKHUIS *et al.*, 2011), where mailing lists were used for in-depth analysis of clinical messages applying natural language processing methods - similar to the ones used in this study. This work proves the power and the significance of the mailing lists by characterizing them as a virtual community of practice that serves as an information hub with easy access to expert advice and opportunities for social networking.

3 METHODOLOGY

In this section, we describe in detail the methodology (and tools) that we have used in our study. We have intended to conceive a general methodology, potentially applicable to any FOSS project.

3.1 DATA EXTRACTION

In order to extract data from the mailing list, we used a mailing list analyzer software called MailingListStats[1] developed by the GSyC/Libresoft research group[2] and available as free software. MailingListStats is a command line based tool that downloads the mboxes to a directory where a database will be created. This database stores all the information contained in the e-mails. Information is retrieved and stored in a monthly basis.

After the data extraction, preprocessing of the data is the next step of our model. Preprocessing is very important because the text of the message body differs from text in articles, books or even spoken language. E-mail text, especially the one in FOSS projects, includes many idiosyncratic uses, such as URLs, terminal commands, Linux distribution names, system paths, Linux/Unix terminal commands, packages, repository names, PGP keys and signatures. It is necessary to preprocess and normalize the text. Generally, in natural language processing practices, after the preprocessing stage, the text is tokenized for later processing. For more details of the data extraction and processing, the entire workflow of our work is represented in Figure 1.

1 Mailing List Stats tool: http://metricsgrimoire.github.com/MailingListStats/

2 LibreSoft Research Group: http://www.libresoft.es/

doi:10.5329/RESI.2014.1302006

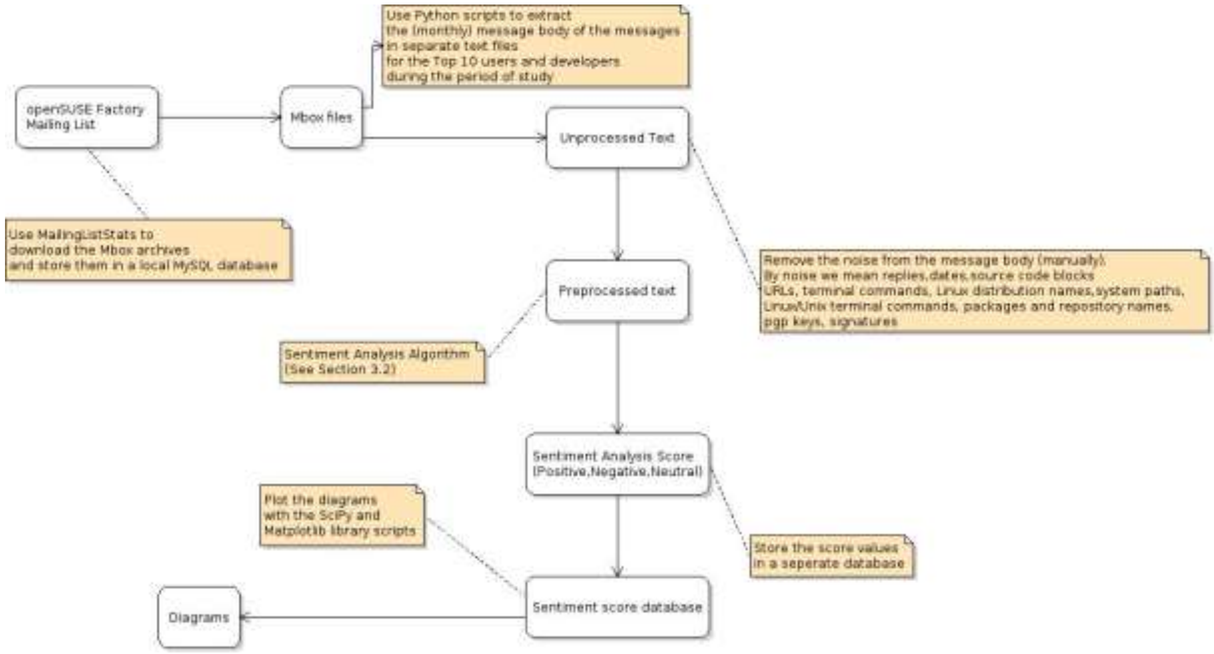

Figure 1. Workflow diagram of our work
Source: elaborated by the authors

3.2 SENTIMENT MODEL

As for the software that is used in our work, we have used the Python programming language and the NLTK package[3] which is a very well know sentiment analysis data set analyzer, although other general machine learning methods for sentiment classification can be found as well.

Another method that could be used for sentiment analysis is counting the number of some specific words for their frequency and their valence, i.e., whether they are positive or negative. Although defining an opinion lexicon (a list of positive and negative opinion words) for annotating words and sentences seems to be a good procedure for sentiment analysis, scientists claim this procedure is far from sufficient for accurate sentiment analysis (LIU *et al.*, 2010).

Our work and model is an 'early' empirical study of sentiment analysis because we only apply machine learning and NLP methods. No other linguist method is being applied (PANG AND LEE, 2008). Moreover, until now there is no official lexicon to tag computer science words and separate their meaning from the literal one. Apart from the opinion lexicon, a significant part of the sentiment model is the algorithm which has been used to tag and analyze the sentiment of the text files. The algorithm we have contains a list of positive and negative words[4] in the

---

[3] Natural Language Toolkit: http://www.nltk.org/

[4] Text Sentiment Analysis Tool: https://github.com/athanrous/text_sentiment_analysis/tree/master/dicts

doi:10.5329/RESI.2014.1302006

YAML format[5]. The reader should acquire knowledge in the field of natural language processing[6] and some functions of it (lemmatisation[7], tokenization[8], speech tagging[9] and especially in the case of NLTK library[10]) in order to be able to understand the sentiment analysis algorithm.

The sentiment algorithm that has been used is as follows:

1. We define one (or more) dictionary of words (also called word list). The design of the dictionaries highly depends on the concrete topic where you want to perform the opinion mining. For example, opinion mining about U.S. elections and opinion mining about the release of the latest Android is different. As a result, the positive/negative expressions could be different but the context vocabulary is also distinct. In our case we defined one dictionary for positive words and another one for negative words.

2. We decide the format of the text we are going to analyze and interact with. As our piece of code interacts with text, splitting, tagging, and extracting information from it, there are several ways to define the structure of the text. Concerning the NLP and tokenization methods, we have many options and ways to analyze the text. In our case we assume the following ones:
   - Each text is a list of sentences
   - Each sentence is a list of tokens
   - Each token is a tuple of three elements: a word form (the exact word that appeared in the text), a word lemma (a generalized version of the word), and a list of associated tags.

3. As in the previous step we have decided the structural shape of the processed text, we can start writing some code to read, and pre-process this text. With pre-process we mean some common first steps in NLP, such as: tokenize, split into sentences, and POS tag.

4. The next step is the basic text preprocessing, where the input is the text as a string and the output is a collection of sentences, each of which is again a collection of tokens.

5. As we have a collection of sentences and we are using NLTK, our forms and lemmas will always be identical. At this point of the process, the only tag associated to each word is its own POS tag provided by NLTK.

---

5 YAML Ain't Markup Language: http://www.yaml.org/

6 Natural Language Processing (Wikipedia): http://en.wikipedia.org/wiki/Natural_language_processing

7 Lemmatisation (Wikipedia): http://en.wikipedia.org/wiki/Lemmatisation

8 Tokenization (Wikipedia): http://en.wikipedia.org/wiki/Tokenization

9 Tagging (Wikipedia): http://en.wikipedia.org/wiki/Part-of-speech_tagging

10 Natural Language Toolkit: http://nltk.org/book/ch05.html

6. The next step is to recognize positive and negative expressions. To achieve this, we use dictionaries, i.e., simple files containing expressions that will be searched in our text.
7. The recognition of positive and negative expressions is not enough for opinion mining. We have to tag the preprocessed text with the dictionaries defined before. Note that while tagging the text, the input is the previously preprocessed text, and the output is the same text, enriched with tags of type "positive" or "negative".
8. The last step is the sentiment measurement of the sentiment tagged text. In our case we count how many positive and negative expressions we detected. For each 'positive' tag we measure it with '> 0', negative with '<0' and neutral or no text found with '= 0' (see Table 1). To see the sentiment score of our text we first aggregate the negative, positive and neutral tags found in the text. We have created a procedure to distinguish situations where the aggregated value is similar, but the balance between positive and negative sentiments is not.

Table 1: Weights assigned to each metric

Score	Classification
> 0	Positive
= 0	Neutral or no text
< 0	Negative

Source: the authors

All the scores for each developer and for each month were stored in a new database table with the following columns: name (e-mail address), date, and score. This database was created in order to display and visualize our data. The source code of the algorithm is publicly available[11].

3.3 SENTIMENT PROPENSITY

The sentiment score per contributor does not provide information about the negativeness and positiveness of a contributor. If, for instance, we had an scenario with contributor A with 300 positive score and 100 negative score and contributor B with 220 positive score and 20 negative score, both would yield the same result in the previous analysis.

Therefore, we have used a new term so as to feature with consistency the positiveness and negativeness of the sentiment score, the *propensity to sentiment*. The propensity is calculated as follows:

Propensity to positive sentiment: $Pr_{pos} = PosScore/NegScore$

Propensity to negative sentiment: $Pr_{neg} = NegScore/PosScore$

Considering the calculation of Pr_{pos} and Pr_{neg} for the previous

[11] Basic Sentiment Analysis Tool: https://github.com/fjavieralba/basic_sentiment_analysis

scenario, we would obtain the following results:

Contributor A: $Pr_{pos} = 300/100 = 3$; $Pr_{neg} = 100/300 = 0.33$

Contributor B: $Pr_{pos} = 220/20 = 11$; $Pr_{neg} = 20/220 = 0.09$

The propensity calculations provide thus a way of normalizing the sentiments of a contributor, not depending on the amount of total score. This can be easily understood from the example above, as even if Contributor A and Contributor B have the same sentiment analysis score of 200, Contributor B has a more positive profile. By taking into consideration the amount of positive versus negative sentiments for an individual, we have a new measure that provides the propensity to positive/negative sentiments.

3.4 DISPLAY AND VISUALIZATION

Display and visualization of our data is a significant part of our study. Without displaying the mined data it is impossible to evaluate, to confirm or decline the assumptions, and to extract any piece of information regarding contributors' sentiments. For displaying our data we have used the Python programming language, combined with scientific libraries (Matplotlib[12], SciPy[13]).

4 CASE STUDY: OPENSUSE FACTORY

Factory is built in its own *openSUSE:Factory* project on the openSUSE instance of the Open Build Service[14]. This is a huge repository of packages. Development, however, does not happen directly in the *openSUSE:Factory*, but in so-called devel projects. A devel project is a project where development happens for a specific group of packages, like multimedia, GNOME, KDE or Kernel. The relation of packages in the openSUSE:Factory project to packages in the devel projects is expressed in the meta data of the packages inside openSUSE:Factory.

Each devel project has its own set of processes, rules and communication channels that fits it best. The reference point for this information is the project description of their Build Service project. Devel projects are also subject to change because the world of FOSS is constantly evolving. Certain software becomes obsolete, standards and defaults change, among others. That means devel projects can change names, get dropped, be newly created, or change content and direction, as can packages in devel projects.

The Factory project follows its own rules and roadmap without disturbing the official openSUSE release. Apart from building software,

---

[12] Matplotlib: http://matplotlib.org/

[13] Scipy: http://www.scipy.org/

[14] OBS – OpenBuildService: http://openbuildservice.org/

contributors and users of the openSUSE Factory do have other kinds of responsibilities.

In this article we study three periods of releases, the period after the openSUSE 11.3 release until the openSUSE 12.2 release (see Roadmap[15]). In other words we analyze the openSUSE Factory developers and users' sentiments for three main releases. These three periods have not been chosen by chance, but based on the fact that the release cycle for the 12.2 release has been postponed for almost three months.

We used the openSUSE Factory mailing list archives from July 2010 to September 2012, totaling 27 months. Mailing list archives are available in mbox format and each mbox file includes messages for one month[16]. In total, we have mined 17,470 messages and analyzed 4,176 messages from 270 mbox files. The last release (12.2) was released September 5th 2012, so we included only the messages for the first 5 days of September 2012. The release periods under consideration in this paper are thus: 11.3 to 11.4 (first period), 11.4 to 12.1 (second period) and 12.1 to 12.2 (third period).

We have studied the top 10 contributors to the openSUSE Factory mailing list. We have selected only ten people, because they are responsible for the majority of the activity in the mailing list; this is known in the literature as the core group (MOCKUS *et al.*, 2002). The study of the rest of contributors is beyond the analysis of this paper, as many contributors exist with low activity, which may have as a consequence situations not taken into account in our methodology.

5 RESULTS

This section contains the results obtained from applying the aforementioned methodology to our case study.

5.1 SENTIMENT ANALYSIS

Figure 2 presents the total sentiment score for each of the top 10 contributors during the complete period under study. As it can be observed, seven of the contributors have a clearly positive score, while only two of them have a negative one; in one case, coolo, the resulting score lies in the low positive values, so we could consider it as neutral. In summary, the global sentiment score as a group is positive, although the variations are high.

---

[15] OpenSUSE Roadmap: http://en.opensuse.org/openSUSE:Roadmap

[16] OpenSUSE Factory Mailing list: http://lists.opensuse.org/opensuse-factory/

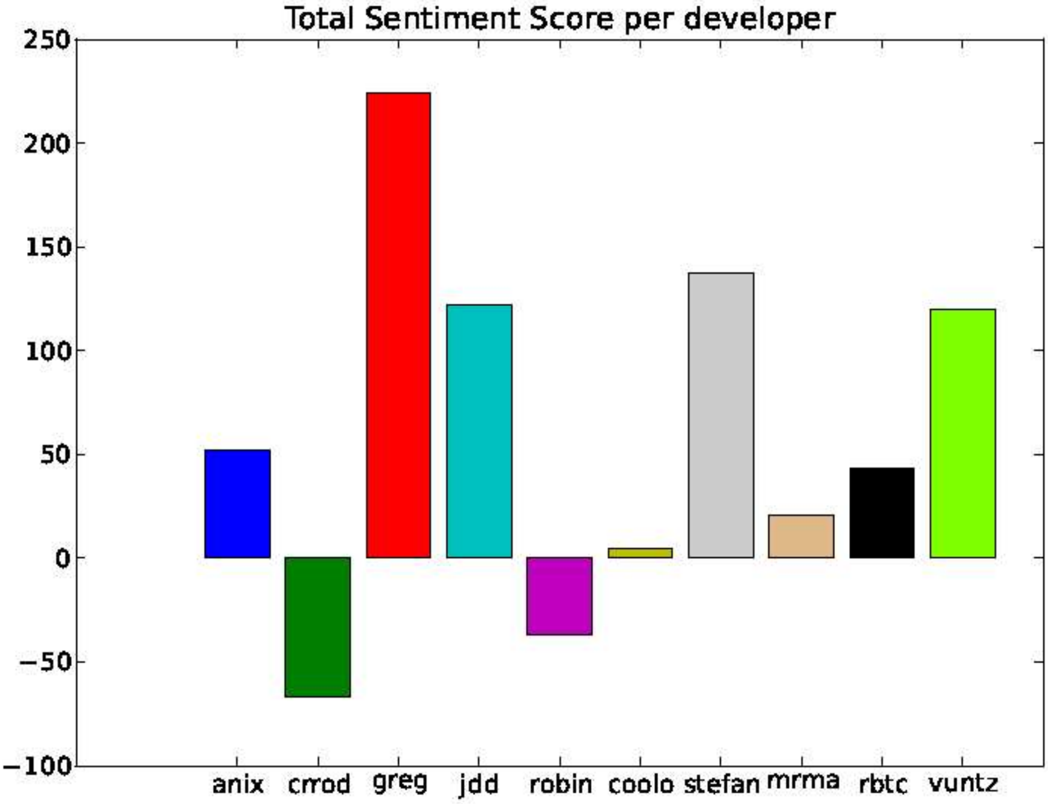

Figure 2. Total sentiment score for each of the top 10 contributors during the complete period under study.
Source: elaborated by the authors based on the results of the study

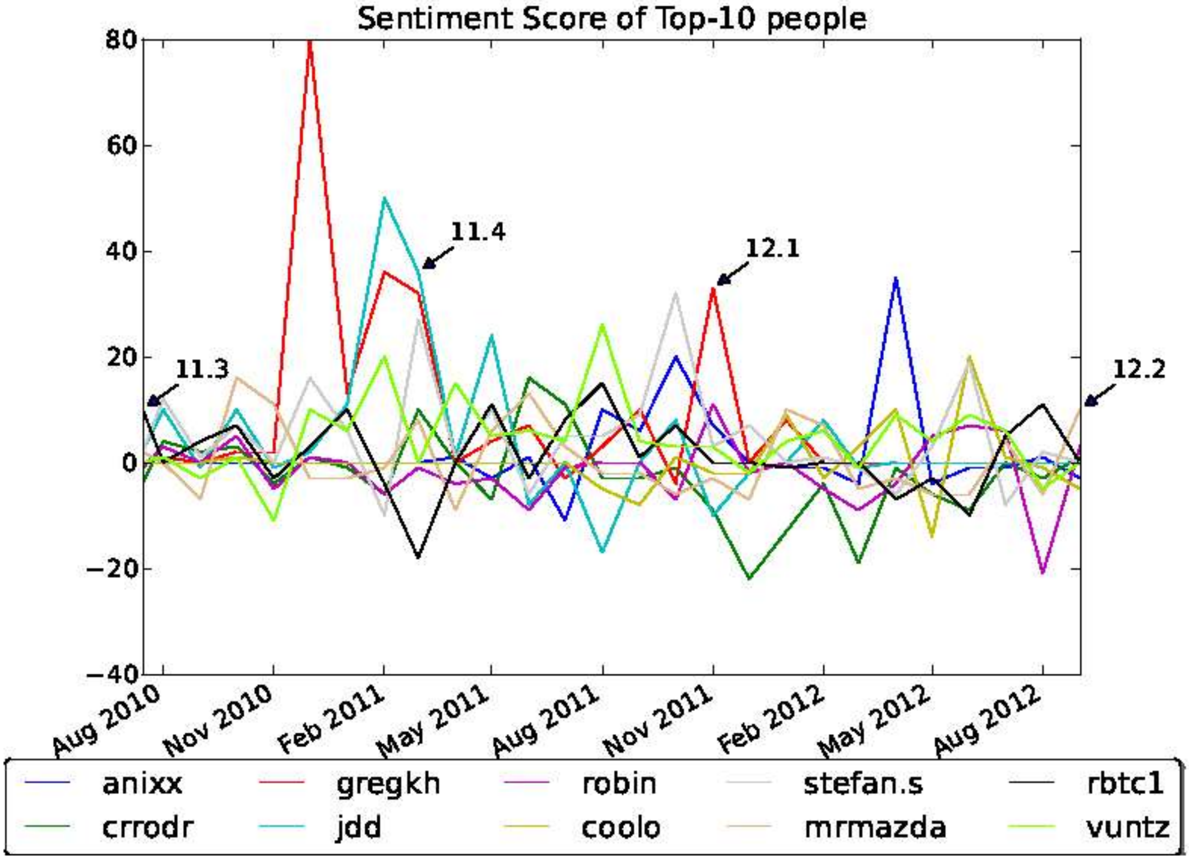

Figure 3. Evolution of the sentiment score for the top 10 contributors during the complete time period under study.
Source: elaborated by the authors based on the results of the study

Figure 3 provides further insight, by showing the evolution of the sentiment score during the 27 months under study. The release times have been annotated in the figure. We can see that sentiment scores vary significantly over time, having in general more extreme values just before a release. The months after a release have, in general, low values compared to the former ones.

Noteworthy is the fact that most developers have had positive and negative months during the 27 months under study, although in general again the number of positive months outweighs the number of negative months. The figure also shows (positive and negative) peaks for developers, and while in some periods we can find synchronized patterns (for instance around February 2012, there is bad mood in the project), there are many cases where we see that some developers have a positive peak while others have a negative one during the same month (for instance, in August 2011 *vuntz* shows a very positive behavior, while *jdd* is significantly negative). We can therefore conclude that there may be general, project-own patterns, but that individual situations also exist and can be observed with our methodology.

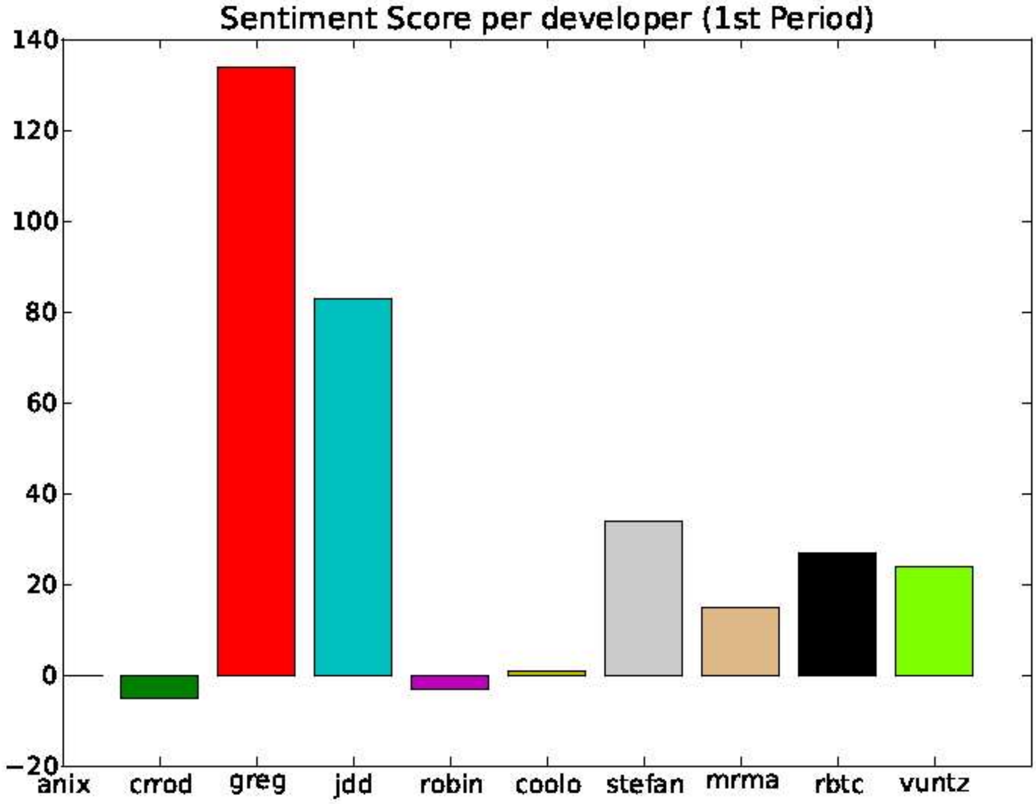

Figure 4. Sentiment score per contributor for the first period.
Source: elaborated by the authors based on the results of the study

First period: If we focus on the sentiment score for the first period, we can observe that this period is being characterized by high differences among developers (see Figure 4). A majority of developers have a positive sentiment, and those who show negative sentiments have very low values of it. However, we can also see that only three developers outreach the score of 20, meaning that the positive values are not strong.

Considering the evolution during this period, two developers (*jdd* and *stefan.s*) raise their score during the months close to the 11.4 release. The lowest score in a month during this period is given by *rbtc1*, shortly before the release of 11.4, as s/he obtains a -18. The high variability in sentiment score during a period is given by the fact that even if *rbtc1* has the lowest value for one month, his total score during the complete period is positive.

Second period: As for the second period, a majority of developers have a positive sentiment after the 11.4 release (see Figure 5). During this period only nine of the ten developers are active in the openSUSE Factory list (*anixx* did not post messages during this period). Just after the release only three developers (*robin*, *mrmazda*, *crrodrig*) have negative sentiments.

By having a look at Figure 5, we see that *mrmazda* and *crrodrig* do not surpass 20 (as a score) and *robin* has the lowest sentiment score (same as *coolo*) for this period of time. Close to the 12.1 release time (Sept 2011 -

Nov 2011), five developers feel positive as their score increases (we could observe the same effect in the first period), but four of them do have negative sentiments during the release time.

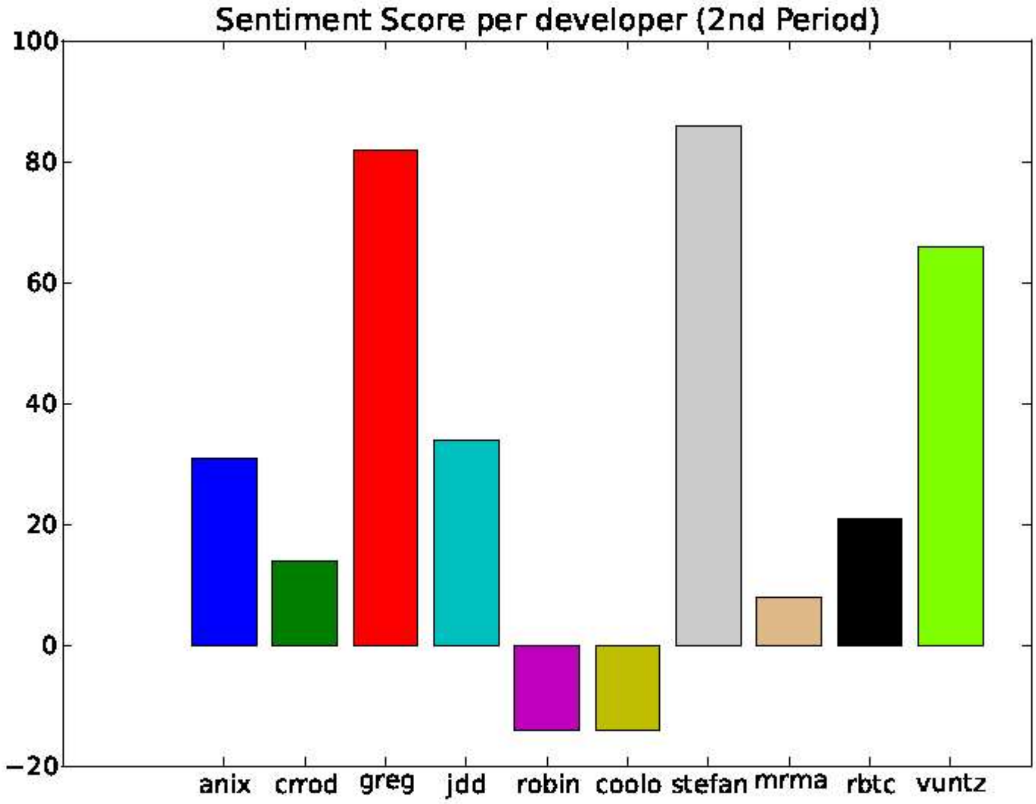

Figure 5. Sentiment score per contributor (second period)
Source: elaborated by the authors based on the results of the study

Only one developer has neutral sentiments, *rbtc1*. Furthermore, it has to be mentioned that, during the time span close to the 11.4 release, the same amount of developers feel positive about the new release (five developers), three developers have a neutral sentiment and only one seems not be negative for the new release.

Seven developers surpass the score limit of 20, and only one has a score between 0 to 10 (*mrma*). Under these circumstances, it turns out that the sentiment scores are higher than in the first period, which means that developers felt more positive with the 11.4 release.

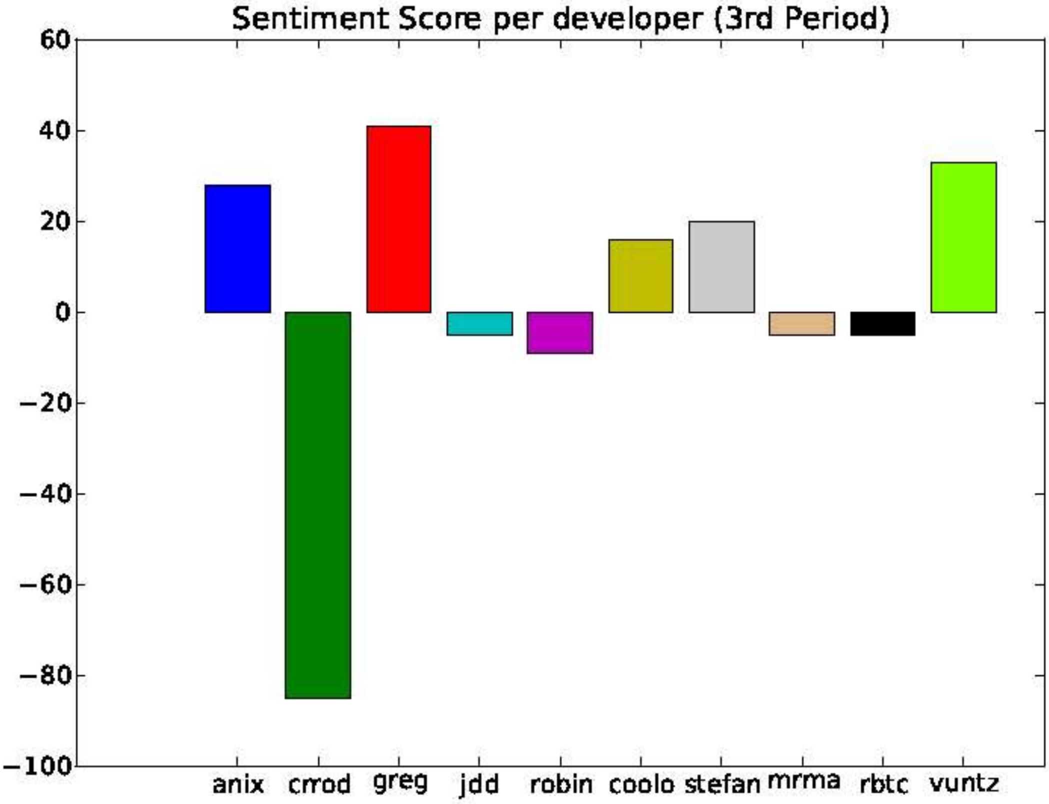

Figure 6. Sentiment scores per contributor for the third period.
Source: elaborated by the authors based on the results of the study

Third period: The third period allows us to see how developers are affected by a negative situation, the delay of a release. It should be noted that delaying a release produces often tensions in the development team, as the project may have dependencies with other projects that are not met. Hence, a delay is a situation that most developers would like to avoid, even in the FOSS world.

That such a situation affects the team is confirmed by the results shown in Figure 6. As we can observe, five of the developers had negative sentiments. Furthermore, *crrod* gets the lowest sentiment score (-85) for all the three periods of study during this period.

If we analyze the evolution of the sentiment score within the period, we can observe how in the dates close to the release date (July 2012 - September 2012) a tendency to neutrality arises: by the time of the 12.2 release, six of the developers do have neutral sentiments. Compared to previous releases, where only two and one had neutral sentiments, it offers a clear sign of how harmful a delay is to the morale of a team. In the meantime only one developer (*anixx*) had a very high score (35). In summary, it turns out that the last period of our study is characterized by negative sentiments.

5.2 SENTIMENT PROPENSITY ANALYSIS

For the analysis of the sentiment propensity analysis, we will display two plots for each period, one that visualizes the propensity to positive sentiments of the contributors and another one with the propensity to negative sentiments.

First period: Figure 7 and 8 present the results for the positive and negative propensity, respectively. It is important to note that the scale for the vertical axis is different. As already shown with the sentiment score, the developers showed general satisfaction during this period.

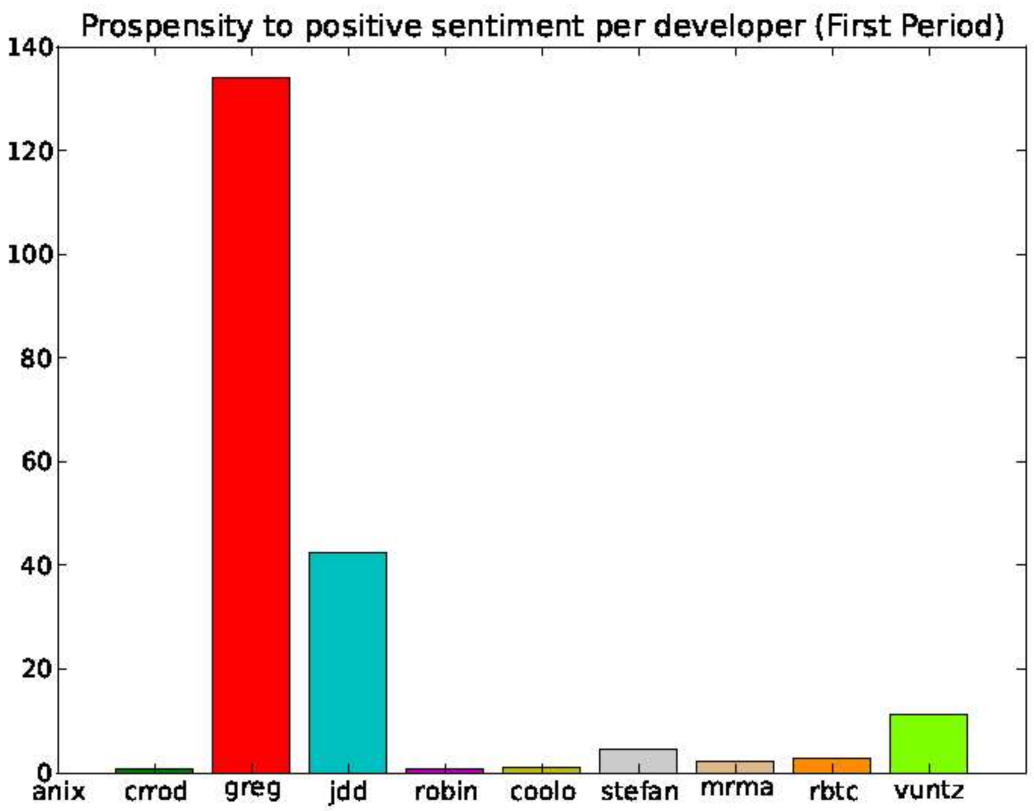

Figure 7. Propensity to positive sentiment per developer (First Period)
Source: elaborated by the authors based on the results of the study

It should be noted that our scores do not take into account the amount of activity performed by the developers. Thus, *coolo* has zero propensity because during the first period he posted messages only during one month. As he shows positive sentiment score, his Pr_{neg} is therefore 0. As well as *coolo*, *greg* has no Pr_{neg} . But this result is not because of not posting messages, but due to the fact that there is no negative score related to *greg* for the first period, where he was very positive.

doi:10.5329/RESI.2014.1302006

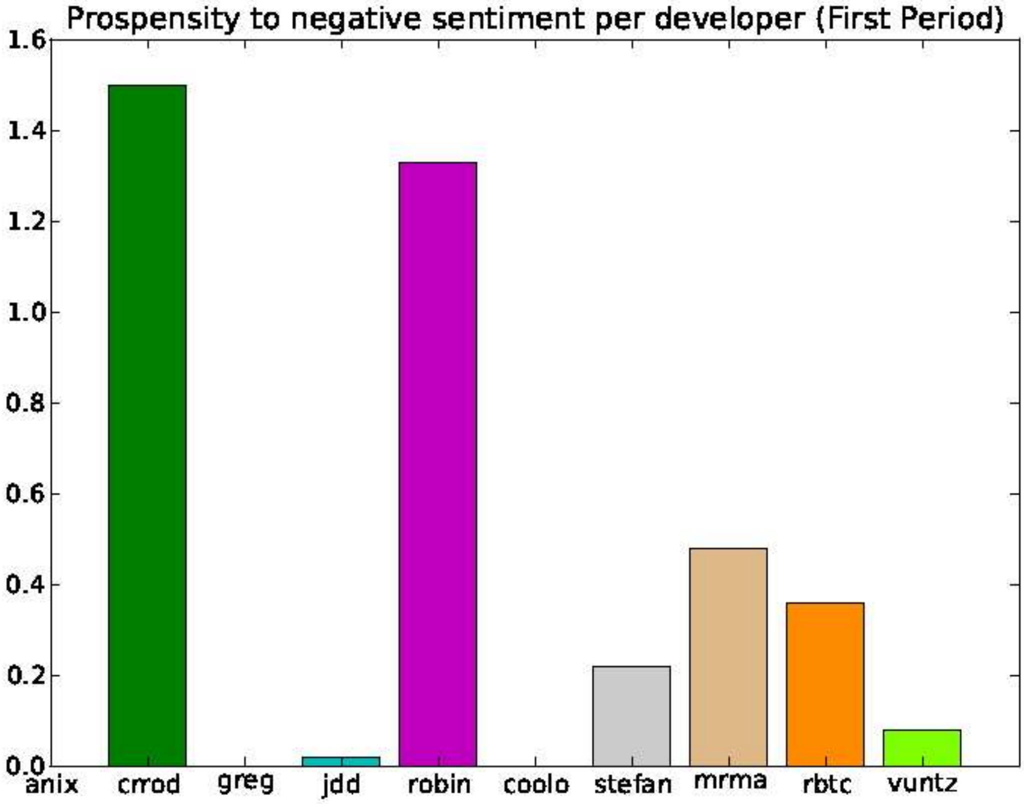

Figure 8. Propensity to negative sentiment per developer (First Period)
Source: elaborated by the authors based on the results of the study

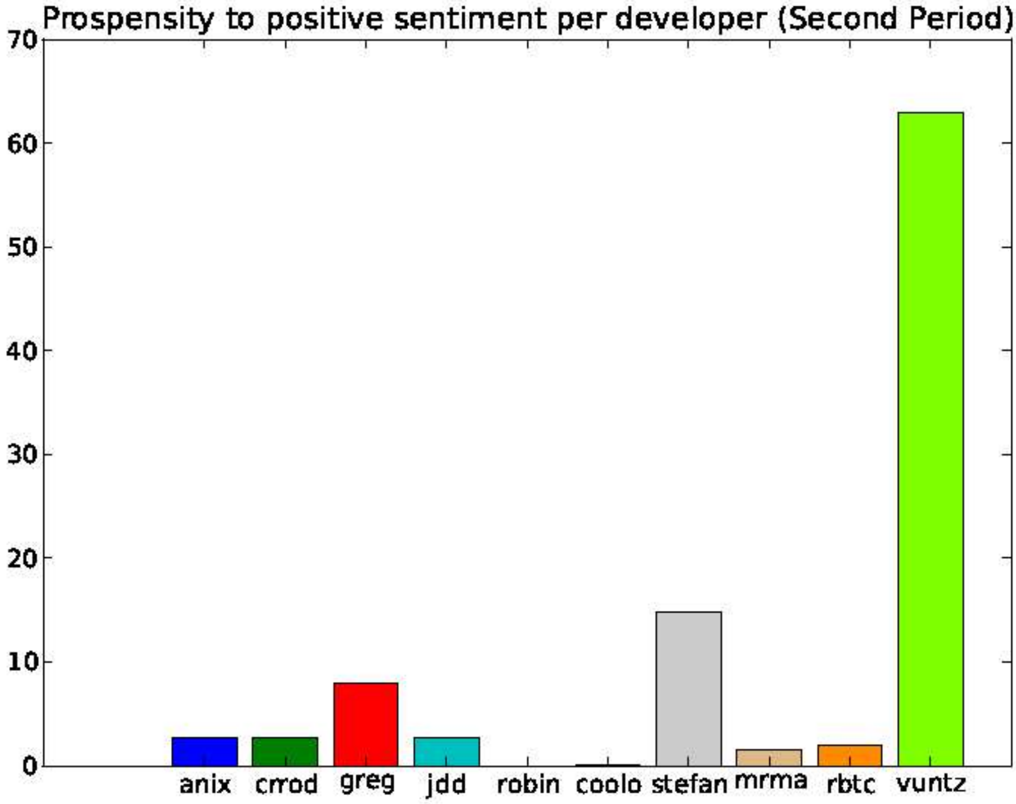

Figure 9. Propensity to positive sentiment per developer (Second Period)
Source: elaborated by the authors based on the results of the study

Second period: The second period gives us more perspective for our analysis. Again, the scale in Figures 9 and 10 differs, but in this period the difference is not of two orders of magnitude, but just about a third.

It is in this period where we start to obtain valuable information from the propensity measures. Developers may have had high values of global positive or negative sentiments because our methodology could extract many sentiment measures from their e-mail messages. With propensity, we obtain values that do not depend that much on the amount of messages but on the balance of positive versus negative feelings. Hence, in this period we can see how those developers with high positive sentiments (and in the first one as well) still have almost no Pr_{neg}. However, it is noteworthy to point out that their Pr_{pos} has decreased considerably.

The propensity measures show that those developers that had negative sentiments during this period showed these negative sentiments frequently and in an absolute manner, i.e., they did not show positive sentiments during this period. For the rest of developers, besides *vuntz* who seems to be very positive during this period, positive and negative propensity have very low values. We could understand that they had mixed feelings during this period. Interestingly enough, the absolute scores showed us a more positive picture. Although we cannot infer it from our findings, we could speculate that the problems (that ended in a delay) that will happen in the

doi:10.5329/RESI.2014.1302006

next period can be traced back to this period. If so, the propensity measures have been indicative of this.

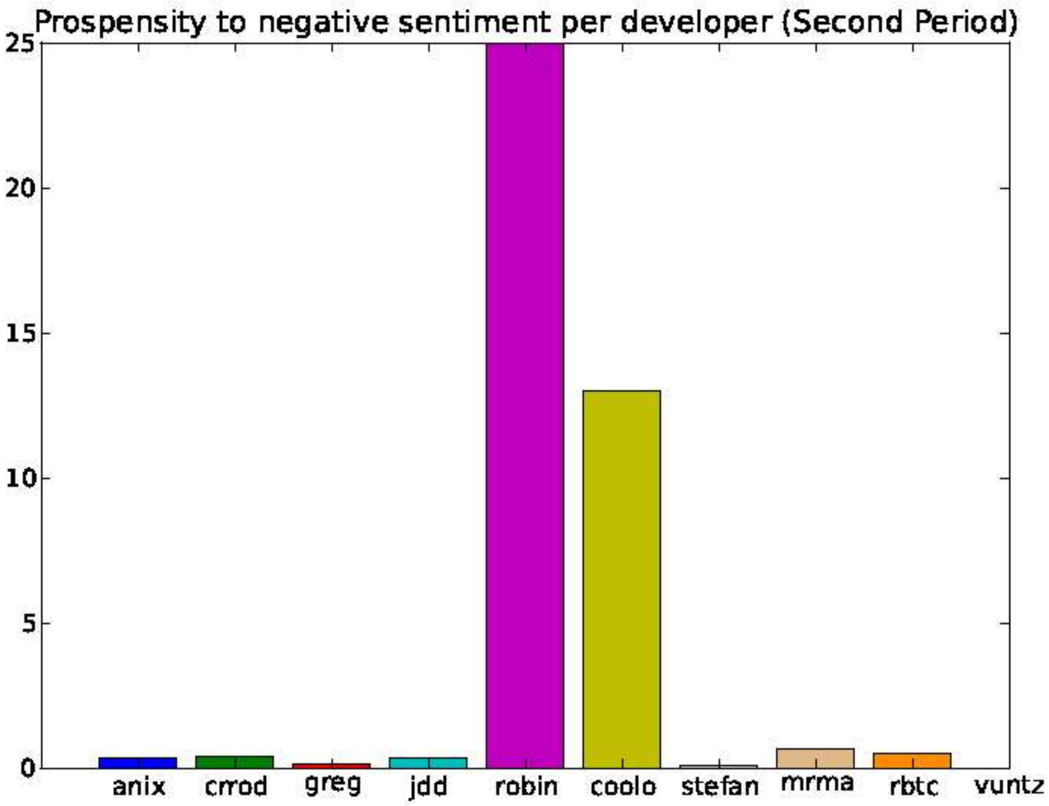

Figure 10. Propensity to negative sentiment per developer (Second Period)
Source: elaborated by the authors based on the results of the study

Third period: The results for the third period, the one that included a delay, can be seen in Figures 11 and 12. If we compare these figures with the one of the total sentiment score (Figure 6), we obtain almost the same information, in this case, although the new figures provide more perspective. So, we can identify *greg* as the positive developer and *crrod* as the developer who has a lot of criticism during this period. The propensity measures for the rest of the developers are very low, meaning that they have mixed feelings. However, on a global basis, the propensity measures are better in this period than in the second one, even if the delay occurred in this third period.

6 CONCLUSIONS AND FURTHER RESEARCH

The aim of this study has been to analyze the sentiment of the contributors in the openSUSE Factory project during a time interval, with three main releases in it. We studied three periods of time in order to mine and analyze a larger sample of data. Our analysis and results show that the openSUSE Factory core contributors felt mostly positive during the time span under study.

The third period showed us that a possible delay in the release cycle of a distribution has a negative effect on the sentiment of the developers, as only two of the developers were positive by the time of release, showing a reduction in the amount of positive thinking developers who contribute to openSUSE Factory project.

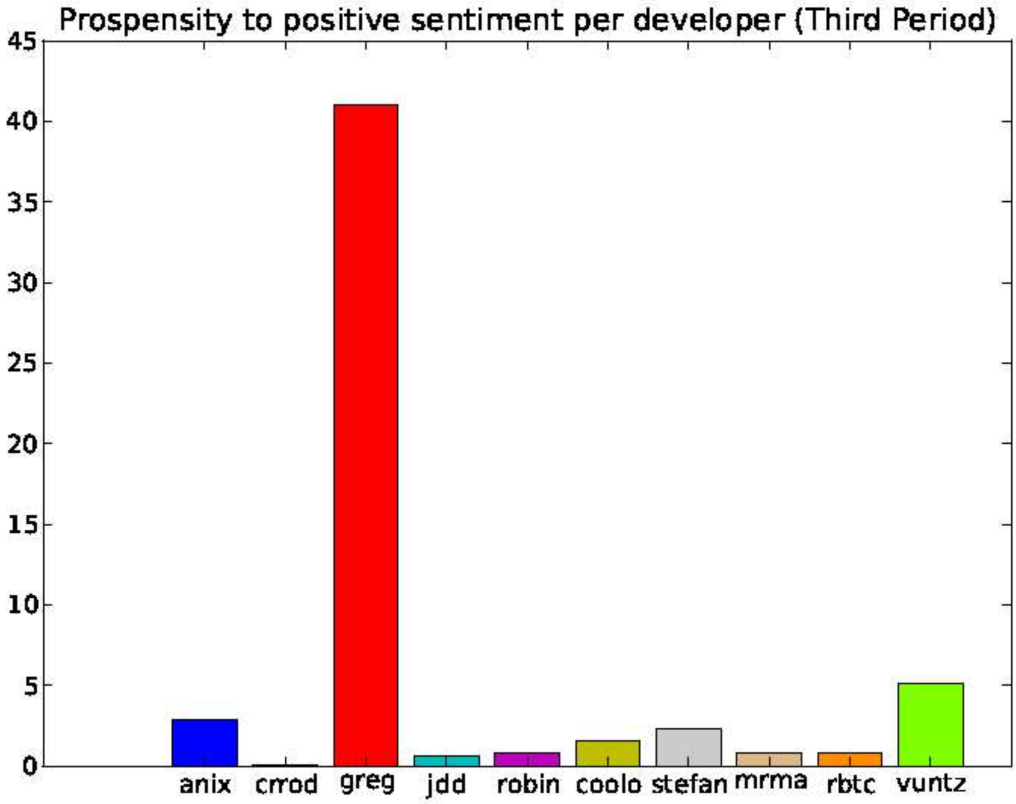

Figure 11. Propensity to positive sentiment per developer (Third Period)
Source: elaborated by the authors based on the results of the study

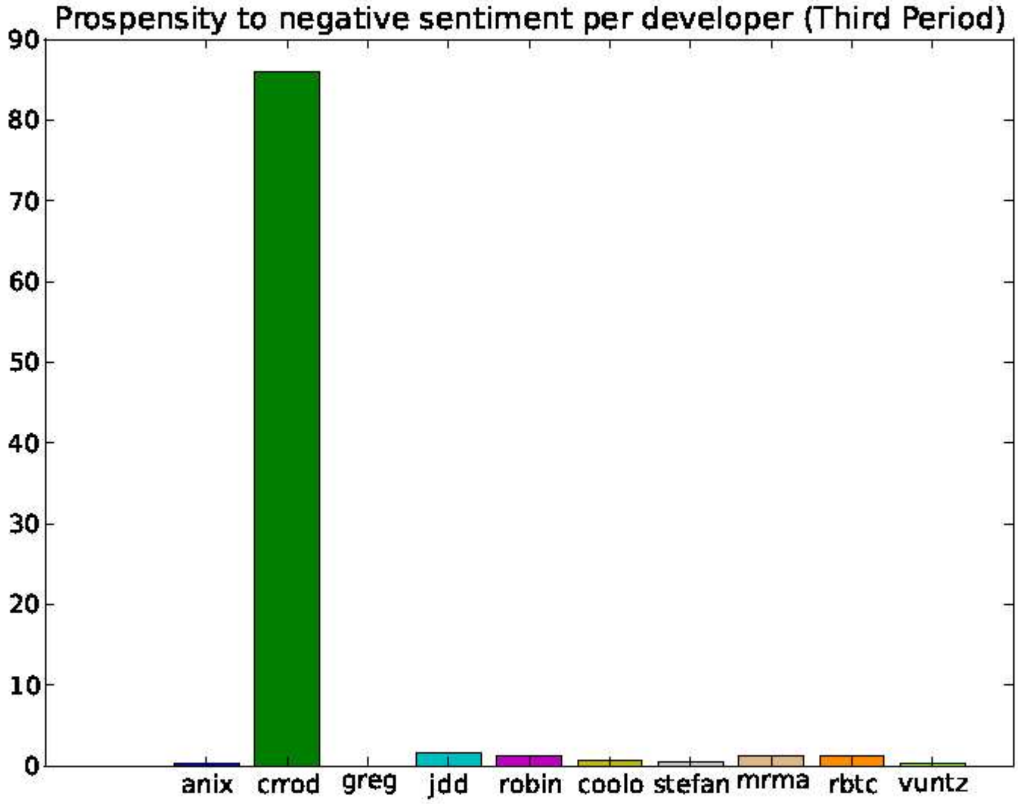

Figure 12. Propensity to negative sentiment per developer (Third Period)
Source: elaborated by the authors based on the results of the study

With the definition of the term "propensity to sentiment", we focused and examined the evolution of the sentiment of the contributors from a different angle than the one that is given by the absolute score. These measures have provided further insight into the sentiment analysis. So, being the second period mostly positive, we could observe already some hints that may be premonitory of the problems that arose in the third period and that led to the delay.

In the near future, we would like to use sentiment analysis on a broader range of projects and situations, to see how usual development circumstances affect the sentiments of the members of a project. Of special interest would be the study of how sentiments are spread through the community and how measures and methods could be introduced that minimize the effect of negative aspects and maximizes those ones that are positively conceived by developers and users.

Further research should generalize our findings to all developers, not those that are the most contributing ones. This could shed some light about different perceptions among the developers and users, and be used in favor of the future of the project.

The inclusion of other data sources could be another research opportunity. So, IRC and posts to bug-tracking system and commit logs could be added to the texts to be analyzed. An interesting issue would be to include versioning system data and source code to study if the positive or negative situation of a developer affects his/her productivity or the quality of the source.

Finally, it would be interesting to use these techniques in situations where there are conflicts in a community. A good example is its use in the study of forks, especially those that are because of disagreements among the developer community. Another good case study could be how long and harsh discussions (named *flamewars* in the hacker jargon) affect the sentiments (and productivity) of a FOSS project.

All in all, although preliminary, we think our research has shown that this is an interesting topic and we hope to see further research in the future.

7 ACKNOWLEDGMENTS

The authors would like to thank the openSUSE community, for their support and comments on some questions raised related to this paper. Athanasios-Ilias Rousinopoulos has done this work being a master's student at the Free Software Master of the Universidad Rey Juan Carlos. The work of Gregorio Robles and Jesús M. González-Barahona has been funded in part by the European Commission under project ALERT (FP7-IST-25809) and by the Spanish Government under project SobreSale (TIN2011-28110).

REFERENCES

BADER, Brett W.; KEGELMEYER, W. Philip; CHEW, Peter A. Multilingual sentiment analysis using latent semantic indexing and machine learning, ICDMW '11 Proceedings of the 2011 IEEE 11th International Conference on Data Mining Workshops, p. 45-52, 2011.

BAL, Daniella; BAL, Malissa; HOGENBOOM, Alexander; HOGENBOOM, Frederik; FRANSINCAR, Flavius; VAN BUNNINGEN, Arthur. Sentiment analysis with a multilingual pipeline, WISE'11 Proceedings of the 12th International Conference on Web Information System Engineering, p. 129-142, 2011.

BALAHUR, Alexandra; TURCHI, Marco. Multilingual sentiment analysis using machine translation? WASSA '12 Proceedings of the 3rd Workshop in Computational Approaches to Subjectivity and Sentiment Analysis, p. 52-60, 2012.

BEKHUIS, Tanja; KREINACKE, Marcos; SPALLEK, Heiko; SONG, Mei; O'DONNELL, Jean A. Using natural language processing to enable in-depth analysis of clinical messages posted to an Internet mailing list: a feasibility study, *Journal of Medical Internet Research*, v. 13, n. 4, 2011.

BOYD-GRABER, Jordan; RESNIK, Philip. Holistic sentiment analysis across languages: multilingual supervised latent Dirichlet allocation, EMNLP '10

Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, p. 45-55, 2010.

GHOSH, R. A.; GLOTT, R.; KRIEGER, B.; ROBLES, G. Free/libre and open source software: survey and study, FLOSS EU project final report, 2002.

GÎNSCA, Alexandru-Lucian; BOROS, Emanuela; IFTENE, Adrian; TRABANDAT, Diana; PEREZ, Cenel-Augusto. Subjectivity and sentiment analysis, WASSA ’11 Proceedings of the 2nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis, p. 189-195, 2011.

HANGAL, Sudheendra; LAM, Monica S. Sentiment analysis on personal email archives. In: Proceedings of the 24th annual ACM Symposium on User Interface Software and Technology, 2011. Available at: http://mobisocial.stanford.edu/papers/informatics11.pdf. Accessed: 27 Dec 2013.

HERTEL, G.; NIEDNER, S.; HERRMANN, S. Motivation of software developers in Open Source projects: an Internet-based survey of contributors to the Linux kernel, *Research Policy*, v. 32, n. 7, p. 1159-1177, 2003. http://dx.doi.org/10.1016/S0048-7333(03)00047-7

LEE, Younggue BaeHongchul; ANAM-DONG, Seongbuk-gu. Sentiment analysis of twitter audiences: measuring the positive or negative influence of popular twitterers. *Journal of the American Society for Information Science and Technology*, v. 63, n. 12, p. 2521-2535, 2012. http://dx.doi.org/10.1002/asi.22768

LIU, Bing; INDURKHYA, N.; DAMERAU, F. J. *Sentiment analysis and subjectivity*. 2. ed., Chapman and Hall/CRC, 2010.

MOCKUS, A.; FIELDING, R. T.; HERBSLEB, J. D. Two case studies of open source software development: Apache and Mozilla. *ACM Transactions on Software Engineering and Methodology* (TOSEM), v. 11, n. 3, p. 309-346, 2002. http://dx.doi.org/10.1145/567793.567795

PALTOGLOU, Georgios; THELWALL, Mike. Twitter, MySpace, Digg: unsupervised sentiment analysis in social media, *ACM Transactions on Intelligent Systems and Technology* (TIST), v. 3, n. 4, article n. 66, September 2012.

PANG, B.; LEE, L. Foundations and trends in information retrieval. *Opinion mining and sentiment analysis*, v. 2, n. 1-2, p. 1-135, 2008.

SAIF, Hassan; HE, Yulan; ALANI, Harith. Semantic sentiment analysis of twitter. ISWC’12 Proceedings of the 11th International Conference on the Semantic Web. Part I, p. 508-524, 2012.

WANG, Hao; CAN, Dogan; KAZEMZADEH, Abe; BAR, François; NARAYANAN, Shrikanth. A system for real-time Twitter sentiment analysis of 2012 U.S. presidential election cycle. ACL ’12 Proceedings of the ACL 2012 System Demonstrations, p. 115-120, 2012.